Check for updates

METHOD ARTICLE

REVISED A lightweight computational method for monitoring lexical response convergence in LLM-based synthetic populations

[version 2; peer review: 2 approved with reservations]
Previously titled: "A lightweight computational method for monitoring response convergence in LLM-based synthetic populations"

Rodrigo Alvarez[1], Mina Soroberto[1], Fernando De la Prieta[1], Mario Cartón[2]

[1]Research Group, University of Salamanca (PIC 999846610),Edificio Multiusos I+D+i, Calle Espejo s/n, Salamanca, 37007, Spain
[2]MADISON MK, Edificio Madison Arena, C/ Enrique Cubero 9, Valladolid, 47014, Spain

v2 **First published:** 30 Jun 2026, **6**:223
https://doi.org/10.12688/openreseurope.24356.1
Latest published: 08 Aug 2026, **6**:223
https://doi.org/10.12688/openreseurope.24356.2

Abstract

Background

LLM-based synthetic populations are increasingly used to simulate open-ended survey, interview, user-research, and group-style responses. These workflows can support early exploratory research, but they also create a methodological risk: agents configured as different participants may nevertheless converge towards similar wording, examples, or answer templates. Without an explicit monitoring step, apparent agreement can be difficult to distinguish from model-level homogenization.

Methods

We present a lightweight computational method for monitoring lexical response convergence in generated agent populations. The method uses a platform-independent response-record schema, deterministic normalization, pairwise lexical overlap, vocabulary-diversity diagnostics, data-quality counters, threshold-based inspection flags, and reproducible JSON reporting. The reference implementation is the Python package synthetic-response-metrics version 0.2.5, which supports CSV, JSON, JSONL, API, and command-line execution.

Open Peer Review

Approval Status ? ?

	1	2
version 2 (revision) 08 Aug 2026		? view
version 1 30 Jun 2026	? view	

1. **AKIN OLAWALE OLUWADAYISI**, Wesley University of Science and Technology Ondo (Ringgold ID: 398081), Ondo, Nigeria
2. **Oleg Illiashenko**, Leeds Beckett University, Leeds, UK

Any reports and responses or comments on the article can be found at the end of the article.

Worked examples and initial validation

We provide initial validation using controlled synthetic examples created for this article. The validation includes a data-quality example, low-, moderate-, and high-convergence scenarios, a population-size sensitivity analysis, and a high-convergence diagnostic pair. These examples show expected metric behaviour, including increasing overlap under controlled convergence, changing pair counts as population size increases, and traceability of alerts through top similar pairs. In addition, an applied example generated from a synthetic-population platform uses 10 synthetic agents and three open-ended questions to check operational portability of the same input schema.

Conclusions

The method provides a reproducible lexical baseline for observing whether generated agents are producing overly similar answers. It does not replace semantic evaluation, human baselines, or domain validation, and the default threshold is provisional. However, it offers a practical monitoring layer that can be embedded in synthetic-population studies before more expensive or domain-specific validation is available.

Plain Language summary

Researchers are beginning to use groups of AI-generated agents to explore how different people might answer questions, react to products, or discuss possible decisions. These simulations are easier to inspect when the answers are varied. A practical problem is that different agents can sometimes give answers that sound too similar, even when they are meant to represent different viewpoints.

This article describes a simple method for checking whether generated answers are becoming too alike. The method cleans the text, compares answers to the same question, reports transparent scores, and flags potential lexical convergence when two answers overlap strongly. The examples were created specifically for this article. They compare response sets that are deliberately diverse, moderately similar, and highly similar. They also show what changes when the number of agents increases, because larger populations create more possible response pairs. A further small applied example uses fictional platform-generated synthetic data to show that the same checks can be run on a realistic export.

The method is intentionally lightweight. It does not require another AI model, embeddings, or an external service. Researchers can rerun it from CSV, JSON, or JSONL files and inspect which pair of agents caused an alert. The scores should be read as an early warning system, not as a complete judgment of whether a synthetic population is valid.

Keywords
synthetic populations; LLM agents; response convergence; diversity metrics; lexical similarity; multi-agent simulation; reproducible methods; synthetic market research



This article is included in the Horizon Europe gateway.

Corresponding author: Rodrigo Alvarez (ralvarezmartin@usal.es)

Author roles: Alvarez R: Conceptualization, Formal Analysis, Methodology, Software, Validation, Writing – Original Draft Preparation, Writing – Review & Editing; **Soroberto M**: Conceptualization, Formal Analysis, Methodology, Software, Supervision, Validation, Writing – Original Draft Preparation, Writing – Review & Editing; **De la Prieta F**: Conceptualization, Funding Acquisition, Supervision, Writing – Review & Editing; **Cartón M**: Software, Supervision, Validation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This project has received funding from theHorizon Europe Framework Programme under grant agreement No [101135927](A catalyst for EuropeaN ClOUd Services in the era of data spaces, high-performance and edge computing [NOUS]).

Copyright: © 2026 Alvarez R *et al*. This is an open access article distributed under the terms of the Creative Commons Attribution License , which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Alvarez R, Soroberto M, De la Prieta F and Cartón M. **A lightweight computational method for monitoring lexical response convergence in LLM-based synthetic populations [version 2; peer review: 2 approved with reservations]** Open Research Europe 2026, **6**:223 https://doi.org/10.12688/openreseurope.24356.2

First published: 30 Jun 2026, **6**:223 https://doi.org/10.12688/openreseurope.24356.1

REVISED **Amendments from Version 1**

This revised version responds to the peer review report on version 1. The title now specifies "lexical" response convergence, aligning it with the abstract and the scope of the method.

Two additions strengthen reproducibility. A new threshold sensitivity analysis (Table 13) re-tabulates the per-question maximum Jaccard similarity for every validation condition and reports how many questions would be flagged at thresholds of 0.30, 0.50, 0.70 and 0.90, computed from the already-deposited outputs. A new reproducibility map (Table 14) links each results table to its input file and expected output file in the archived release.

Methods now state the computational environment: Python 3.10 or later, no third-party runtime dependencies, and 17 automated tests. The normalization pipeline is documented precisely, including the cleanup pattern and the stopword list sizes (20 English, 21 Spanish, 41 combined). The input data model additionally documents the loader for interaction-style exports.

Data availability now enumerates the complete manifest of the 28 deposited files with their paths inside the archive, states that inputs and expected outputs are both deposited, and explains why the data and the software share a single record.

Limitations have been strengthened. The article now states explicitly that the method cannot detect semantic convergence, distinguishes dispersion of outputs from fidelity of agent conditioning, and acknowledges that the method is platform-independent but not export-independent. Agent-conditioning fidelity is added to the planned extensions (Table 16). The Conclusions separate what the examples demonstrate from what they do not.

The Acknowledgements now carry the required funding statements for the NOUS (101135927) and MADSYNTHESIS AI (TSI-100130-2024-0134) projects.

Former Tables 13 and 14 are renumbered 15 and 16. No reported result changed.

Any further responses from the reviewers can be found at the end of the article

Introduction

Synthetic populations generated with LLM agents are becoming a practical tool for exploratory social research, user research, public-policy prototyping, education, and market-research workflows. Prior work has shown how language models can be used to simulate human samples, replicate selected human-subject study patterns, support interactive generative-agent simulations, and assist marketing or consumer-research ideation (Argyle et al., 2023; Aher et al., 2023; Park et al., 2023; Arora et al., 2025; De Freitas et al., 2025; Toubia et al., 2025). In these settings, researchers define synthetic participants or population segments and ask agents to produce open-ended responses to prompts. The prompts may resemble survey questions, interviews, product-reaction tasks, deliberation exercises, or group-style discussions. This workflow can make early-stage exploration faster and cheaper, but it introduces a new measurement problem: generated answers are mediated by a shared model, shared prompting environment, and shared generation process.

The resulting outputs can appear coherent and fluent while still being methodologically fragile. A research team may configure agents with different demographic, behavioural, or preference attributes, yet the generated answers may reuse similar vocabulary, examples, argument order, or rhetorical templates. This concern connects with evidence that generative AI can homogenize open-ended responses and reduce collective content diversity in some settings (Anderson et al., 2024; Doshi and Hauser, 2024; Zhang et al., 2025). If outputs are reviewed manually, local convergence can be missed because the responses remain plausible. If outputs are summarized into aggregate themes, local near-duplicates may disappear from view. For synthetic-population studies, response convergence is therefore both a validity concern and an observability concern.

This issue is related to broader concerns about homogenization in generative AI, model collapse, synthetic-text feedback loops, and the evaluation of LLM-based agents (Guo et al., 2024a; Shumailov et al., 2024; Chang et al., 2024; Guo et al., 2024b; Mohammadi et al., 2025). It is also distinct from those topics in one important respect: researchers using synthetic populations often need a lightweight operational signal during study design. They need to know whether a prompt, population design, or model configuration is producing responses that are lexically too similar before they invest in deeper semantic analysis, human comparison, or domain-specific review.

The method proposed here treats response-convergence monitoring as a deterministic post-processing step. It does not claim to measure the full validity of a synthetic population. It does not infer whether an answer is true, representative, persuasive, or human-like. Instead, it focuses on a narrower and reproducible question: when several generated agents answer the same prompt, how much lexical overlap is present, where are the local extremes, and which response pairs should be inspected?

A lexical baseline is useful because it is transparent, cheap to compute, and easy to reproduce. Set-overlap measures such as Jaccard similarity are long established for comparing token sets, entropy provides an interpretable measure of concentration, distinct-n metrics have been used to monitor diversity in generated text, and automatic text-generation evaluation has long considered overlap and degeneration trade-offs (Jaccard, 1901; Shannon, 1948; Papineni et al., 2002; Manning et al., 2008; Li et al., 2016; Holtzman et al., 2020). Unlike embedding-based similarity, lexical overlap can be recalculated without model access or hidden external dependencies. Unlike manual review, it can be applied consistently across many runs. It is also interpretable: a high maximum similarity can be traced to a concrete pair of responses, and a low diversity score can be connected to repeated tokens or phrases.

The contribution of this Method Article is threefold. First, it defines a platform-independent data model and algorithmic workflow for monitoring response convergence in generated agent populations. Second, it provides an open-source reference implementation with data-quality counters, duplicate handling, pairwise similarity distributions, and optional pair-level diagnostics. Third, it documents article-specific controlled examples and initial validation scenarios that show how the method behaves under known convergence conditions. The article also includes a small platform-generated applied example to illustrate portability to a normalized export without making the method dependent on that platform.

Methods

The proposed method can be applied after a synthetic-population run or embedded inside a generation pipeline. The only required input is a set of generated answers with a question identifier, an agent identifier, and response text. Optional metadata such as run identifier, model, temperature, population segment, or timestamp can be preserved for later grouping and audit. The method is deliberately independent of model internals, sampling probabilities, embedding services, and orchestration frameworks.

Method design requirements

The method was designed around five requirements, following common expectations for reproducible computational research and research-software citation (Sandve et al., 2013; Smith et al., 2016; Wilkinson et al., 2016; Wilson et al., 2017). First, it must be platform agnostic: it should accept ordinary tabular or JSON response exports rather than depend on a specific agent framework. Second, it must be lightweight: the default metrics must run locally and add negligible overhead compared with generation. Third, it must be reproducible: thresholds, normalization settings, input records, and outputs must be exportable and rerunnable. Fourth, it must be interpretable: the metrics should be understandable to research teams without specialist model-evaluation infrastructure. Fifth, it must be extensible: lexical metrics should provide the baseline layer while allowing semantic, syntactic, and human-baseline metrics to be added later.

Framework architecture

The method is organized as a five-stage pipeline: response ingestion, normalization, metric computation, alerting, and reporting (Figure 1). This separation keeps the procedure reusable. Ingestion handles conversion into canonical response records. Normalization defines the text representation used for comparison. Metric computation produces continuous scores. Alerting converts selected scores into inspection flags. Reporting stores results in a stable machine-readable object; the corresponding layer responsibilities are summarized in Table 1.

Input data model

The canonical input unit is a response record (Table 2). The minimum required fields are question_id, agent_id, and text. Optional metadata fields allow researchers to preserve run identifiers, model names, sampling settings, population or segment identifiers, and timestamps. The design is intentionally simple because most generation workflows can export this information without changing their internal state model.

The reference implementation accepts common aliases such as question, content, answer, response, agentId, and participant_id. This reduces adoption friction when response data come from notebooks, dashboards, simulators, or annotation files. The method itself does not require a database schema or a specific application backend. Similarly, exports from external platforms can be analysed after normalization to the required question_id, agent_id, and text fields. Beyond CSV, JSON, and JSONL inputs, the implementation also provides a dedicated loader for interaction-style



Deterministic monitoring pipeline

Figure 1. Deterministic response-convergence monitoring pipeline used by the framework.

Table 1. Framework layers and responsibilities.

Layer	Responsibility	Output
Response ingestion	Load response records or interaction exports and convert them into canonical response records.	List of records with question, agent, text, and optional metadata.
Normalization	Apply deterministic text preprocessing before comparison.	Token sequences and token sets.
Metric computation	Calculate pairwise lexical overlap and vocabulary-diversity metrics by question.	Per-question metric rows.
Alerting	Apply configurable rules to identify questions requiring inspection.	Alert flags and run-level counts.
Reporting	Persist outputs in a machine-readable format for audit and rerun.	JSON object with summary and by_question blocks.

Table 2. Generic response-record schema.

Field	Required	Role
question_id	Yes	Groups responses that answer the same question, task, or prompt.
agent_id	Yes	Identifies the synthetic participant or agent producing the answer.
text	Yes	Contains the generated answer to be evaluated.
run_id	No	Links records to a study execution or batch.
model	No	Stores the model or deployment used for generation.
temperature	No	Stores the sampling temperature or analogous setting.
population_id	No	Links the answer to a population, segment, or scenario.
timestamp	No	Stores generation or export time for traceability.

exports, including runs in which a single interviewer question applies to all agents. Where a platform offers no export capability at all, applying the method requires a small adapter that maps its output onto the three required fields.

Configuration validation and record handling

Configuration values, including the options and defaults in Table 3, are validated before metrics are computed. The maximum-similarity threshold must lie between 0 and 1, the minimum number of answers must be at least two, the minimum token length must be at least one, and duplicate-record handling must be selected from a documented set of policies. These checks are included to prevent silent misconfiguration when the method is used in scripts, notebooks, or automated study pipelines.

The default duplicate policy is latest. This is useful when a pipeline emits intermediate and final answers for the same question-agent pair. Alternative policies allow users to keep the first response, raise an error, or keep all repeated records

Table 3. Core configuration options and defaults in synthetic-response-metrics 0.2.5.

Configuration item	Default	Purpose
threshold_max_similarity	0.70	Provisional alert threshold applied to maximum Jaccard similarity.
min_answers	2	Minimum valid responses required before pairwise metrics are computed.
language	auto	Selects default stopwords for English/Spanish or combined mode.
min_token_length	3	Filters short tokens after normalization.
duplicate_policy	latest	Controls repeated question-agent records: latest, first, error, or keep_all.
include_pairs	False	Controls whether top similar agent pairs are included in the JSON output.

by treating each occurrence as a separate response instance. Regardless of the policy, duplicate_records_count is reported so that repeated records are visible before interpreting the metric values.

Method assumptions

The current method assumes that responses are open-ended text answers and that convergence should be evaluated within each question rather than across unrelated prompts. It also assumes that the researcher is interested in lexical behaviour as an interpretable baseline. The lexical scope is a deliberate design decision rather than an oversight: because all metrics are computed from normalized token sets using only the Python standard library, the reported values are transparent, auditable, inexpensive, and recomputable indefinitely without access to the generating model or to a versioned embedding service. That choice carries a cost which we state explicitly, namely that the method cannot detect semantic convergence. Two agents that express the same opinion, recommendation, or line of reasoning in different vocabulary will produce a low Jaccard score and will not be flagged, and two answers can reuse words while making different claims. The method therefore flags potential lexical convergence and must not be read as evidence of semantic diversity.

The default threshold of 0.70 for maximum Jaccard similarity is provisional. It is a transparent starting point for inspection rather than a universal cutoff, and its behaviour across the article-specific validation conditions is reported in the threshold sensitivity analysis (Table 13). Researchers should report the threshold, the normalization settings, the number of valid answers, and the number of response pairs. These denominators matter because a larger population creates more pairwise opportunities for local overlap.

Algorithmic workflow

The procedure can be described as a deterministic algorithm (Table 4). First, response records are loaded from CSV, JSON, or JSONL. Second, public configuration values are validated. Third, repeated question-agent records are handled according to the duplicate policy. Fourth, records are grouped by question_id. Fifth, each response is normalized using case folding, separator replacement, tokenization, stopword filtering, and minimum token-length filtering. Sixth, empty responses after normalization are excluded from pairwise metric computation while their exclusion is counted. Seventh, all valid answer pairs are generated within each question. Eighth, lexical overlap, containment, duplicate, diversity, and length diagnostics are computed. Ninth, the provisional alert rule is applied. Finally, results are written as a JSON object with summary and by_question blocks.

Normalization pipeline

Each response is normalized before metrics are computed. The default pipeline applies Unicode-aware case folding, replaces every character outside the digits 0-9, the Latin letters A-Z and a-z, the accented Latin range from U+00C0 to U+00FF, and the space character with a space, splits on whitespace, discards tokens shorter than the configured minimum length, and then removes stopwords. The cleanup pattern preserves accented Latin characters, so non-English text is not silently degraded. The default minimum token length is three characters, and the length filter is applied before the stopword filter. The stopword lists are deliberately minimal and ship with the package rather than being inherited from an external natural-language-processing distribution: 20 English terms, 21 Spanish terms, and 41 terms in the combined automatic mode. Both lists are reproduced in full in Extended data and in the archived source file src/synthetic_response_metrics/normalization.py, because convergence scores depend on them. A user-supplied newline-delimited file passed with the –stopwords option fully replaces the language list rather than extending it.

Table 4. Algorithmic workflow for response-convergence monitoring.

Step	Operation	Reproducibility output
1	Load records from CSV, JSON, or JSONL.	records_total and input schema.
2	Validate configuration and duplicate policy.	config block in JSON output.
3	Normalize response text deterministically.	language, stopwords, and min_token_length.
4	Group valid answers by question.	questions_total and questions_evaluated.
5	Generate answer pairs within each question.	answers_count and pairs_count.
6	Compute lexical and diversity metrics.	by_question metrics and run summary.
7	Apply provisional inspection flag.	convergence_alert field, shown as Alert in manuscript tables.
8	Write machine-readable output.	summary and by_question JSON blocks.

Normalization is deterministic and configuration-driven. This matters because convergence scores can change when stopwords, token-length thresholds, or language-specific preprocessing rules change. The method therefore treats normalization settings as part of the reproducibility record. Responses that become empty after normalization are excluded from pairwise metric computation, but the exclusion is counted explicitly as excluded_empty_after_normalization.

Lexical metric suite

The metric suite is lexical, deterministic, and lightweight. For each question with enough valid answers, the method forms all unordered response pairs and calculates Jaccard similarity over normalized token sets. Jaccard similarity is used because it is deterministic, bounded between 0 and 1, and directly inspectable as overlap divided by union (Jaccard, 1901; Manning et al., 2008). The method then reports average, maximum, minimum, median, p90, and standard deviation of pairwise similarity. This distributional view is useful because a run may have low average overlap while still containing one highly convergent pair.

For each question q, let R_q be the set of valid normalized responses and let n_q = |R_q|. Each response r_i is represented by a normalized token sequence T_i and a token set S_i. For each unordered pair (i, j), the method computes J(S_i, S_j) = |S_i ∩ S_j|/|S_i ∪ S_j|and C(S_i, S_j) = |S_i ∩ S_j|/min(|S_i|,|S_j|), where J is Jaccard similarity and C is containment similarity. avg_jaccard_similarity is the mean of J across all unordered pairs; max_jaccard_similarity, min_jaccard_similarity, median_jaccard_similarity, p90_jaccard_similarity, and std_jaccard_similarity are defined analogously over the same pairwise Jaccard distribution. max_containment_similarity is the maximum C value across pairs, and exact_duplicate_count is the number of unordered pairs with identical normalized token sequences.

Diversity and length diagnostics are computed from the normalized token sequences for each question. distinct_1 is the number of unique normalized unigrams divided by the total number of normalized unigram tokens; distinct_2 is the number of unique normalized bigrams divided by the total number of normalized bigrams; token_entropy is Shannon entropy over normalized token frequencies; mean_response_tokens is the mean normalized-token length; and response_length_cv is the coefficient of variation of normalized response length. Operationally, the method collects valid responses, normalizes them into token sequences and token sets, generates unordered pairs, computes pairwise Jaccard and containment, aggregates the pairwise distributions, computes diversity and length diagnostics, and then applies the configured inspection rule.

The reporting layer combines the lexical overlap, containment, duplicate, diversity, and length diagnostics listed in Table 5. Vocabulary-level diversity is reported through lexical_unique_ratio, distinct_1, distinct_2, token entropy, mean response length, and response-length coefficient of variation. In the current implementation, lexical_unique_ratio is equivalent to distinct_1: both are computed as the number of unique normalized unigram tokens divided by the total number of normalized unigram tokens across valid responses for a question. The manuscript retains lexical_unique_ratio

Table 5. Lexical and diversity metrics included in synthetic-response-metrics 0.2.5.

Metric	Computation	Purpose
Jaccard distribution	Average, maximum, minimum, median, p90, and standard deviation of pairwise token-set Jaccard scores.	Summarizes both typical and local lexical convergence.
max_containment_similarity	Intersection size divided by the smaller token set for each response pair.	Flags subset-like overlap when answer lengths differ.
lexical_unique_ratio and distinct_1	Unique normalized unigrams divided by total normalized unigrams.	Reports vocabulary breadth using a descriptive name and a literature-aligned distinct-n name.
distinct_2	Unique normalized bigrams divided by total normalized bigrams.	Captures repeated short phrase patterns.
token_entropy	Shannon entropy over normalized token frequencies.	Captures concentration or spread of vocabulary use.
length diagnostics	Mean response tokens and response-length coefficient of variation.	Separates lexical convergence from length effects.
exact_duplicate_count	Number and proportion of response pairs with identical normalized token sequences.	Identifies direct repetition after normalization.

as a descriptive reporting name and distinct_1 for consistency with the text-generation diversity literature. These metrics provide complementary views of concentration: unique-ratio scores summarize token-type reuse, distinct-n scores capture unigram and bigram diversity, and entropy increases when token use is more evenly spread (Shannon, 1948; Li et al., 2016; Zhu et al., 2018). They are not substitutes for semantic validity, but they help identify response sets dominated by repeated vocabulary or phrasing.

Alerting and reporting

The method separates continuous metric values from alerting rules. Continuous scores are the primary result and should be reported for all runs. Alerts are operational aids that identify questions requiring inspection. The default rule flags a question when max_jaccard_similarity is greater than or equal to the configured threshold_max_similarity, which is 0.70 by default, and at least two valid answers are available. In the JSON output, convergence_alert is the canonical Boolean summary field indicating whether the configured maximum-similarity rule was triggered for at least one evaluated question. Per-question rows use alert for the same rule at question level, and manuscript tables use the shorter label Alert as a human-readable column heading.

When include_pairs is enabled, the output also includes top_similar_pairs for each evaluated question. Each row identifies the two agent identifiers, the original record indexes, Jaccard similarity, and containment similarity. Together with the alert and summary fields in Table 6, this diagnostic is designed for interpretability: an alert should not only tell researchers that potential lexical convergence may be present, but also point to the response pair that triggered inspection.

Reference implementation

The reference implementation is the Python package synthetic-response-metrics. In this article, the package is used to make the method executable and reproducible; the methodological contribution is the monitoring procedure rather than the package alone. The implementation exposes ResponseRecord, DispersionConfig, compute_dispersion, and DispersionResult through the public API, and provides a command-line interface for reproducible examples.

```
srm compute examples/ore_validation/convergence_scenarios/high_convergence.csv
--include-pairs --language en --output examples/ore_validation/
convergence_scenarios/high_convergence_results.json
```

The package currently supports CSV, JSON, and JSONL inputs. It includes command-line options for language, custom stopwords, duplicate handling, pair diagnostics, alert threshold, minimum answers, and minimum token length. The implementation requires Python 3.10 or later and declares no third-party runtime dependencies, because every metric is computed with the Python standard library. This is itself a reproducibility property, since the reported values cannot drift as external similarity or embedding libraries are updated. The archived release includes 17 automated tests, executable with pytest from the repository root, covering the four duplicate-handling policies, exclusion of responses that become empty after normalization, the alert rule applied to max_jaccard_similarity, rejection of invalid configurations, equivalence of CSV and JSON inputs, command-line output structure, error reporting for malformed JSONL lines, and preservation of Unicode characters. The reference implementation, synthetic-response-metrics (Alvarez et al., 2026), is archived as version v0.2.5 at https://doi.org/10.5281/zenodo.20537552 and is available from https://github.com/ralvar-ezmartin/synthetic-response-metrics under the MIT License.

Table 6. Reporting and diagnostic fields formalized in version 0.2.5.

Output field family	Examples	Interpretation
Run summary	questions_total, questions_evaluated, records_total, runtime_ms	Documents scope and reproducibility context.
Data-quality counters	missing_text_count, missing_agent_id_count, duplicate_records_count	Makes input problems visible before metric interpretation.
Pairwise distributions	avg/max/min/median/ p90/std_jaccard_similarity	Shows both central tendency and local extremes.
Diversity diagnostics	distinct_1, distinct_2, token_entropy, mean_response_tokens	Separates repeated wording from broader vocabulary patterns.
Pair-level explanations	top_similar_pairs with agent ids and record indexes	Makes alerts auditable and inspectable.

Article-specific validation and worked examples

Open Research Europe guidance for Method Articles indicates that validation results should be provided where available, and that use cases or worked examples are appropriate when full method validation has not yet been completed. This article therefore uses controlled synthetic examples created specifically for the manuscript. The examples test whether the method behaves as expected under known data-quality, convergence, population-size, and diagnostic conditions. They do not establish domain-specific thresholds and they do not replace validation against human-response baselines.

Validation design

The validation design contains four controlled blocks. The controlled data-quality example tests empty text, duplicate records, exact duplicate pairs, and diverse answers in one compact file. The convergence scenarios compare low, moderate, and high lexical overlap while keeping the number of questions and agents constant. The population-size sensitivity examples increase N from 3 to 5 and 10 while preserving comparable prompts. The diagnostic case creates a local high-overlap pair among otherwise distinct responses. The applied platform-generated example is kept separate from these controlled checks, so expected metric behaviour and operational portability remain distinct concerns (Figure 2).

Synthetic dataset construction

The validation datasets were constructed specifically for this manuscript as controlled examples rather than as samples from a human-response population. Each dataset was designed to exercise a predefined behaviour of the method. The controlled data-quality example includes deliberately inserted missing text, repeated question-agent records, and exact duplicate response pairs. The low-, moderate-, and high-convergence scenarios manipulate lexical overlap while keeping the number of questions and agents constant. The population-size examples vary the number of agents while preserving comparable prompt structure, so that the effect of increasing the number of response pairs can be inspected. The diagnostic case introduces one locally high-overlap pair among otherwise distinct responses.

The datasets were created to support functional checking, reproducibility, and manual inspectability. They should therefore be interpreted as article-specific controlled examples and sanity checks, not as evidence that the default threshold generalizes across domains, populations, models, or prompt types. The archived release contains the input files, expected JSON outputs, and derived table files used to reproduce the manuscript results, with the article-specific datasets listed in Table 7.

All validation data are synthetic and article-specific. The examples are intentionally small so that readers can inspect the inputs manually and rerun the outputs through the command-line interface. This is important for a Method Article: the validation should not only report values, but also demonstrate how another researcher could reproduce the procedure.

Controlled data-quality example

The controlled example contains three questions and four agents. It includes one empty answer cell, one repeated question-agent record, one exact duplicate response pair, and otherwise diverse answers. The purpose is to show that data-quality counters and convergence metrics are reported together in the output (Table 8). A high similarity score should not be interpreted before checking whether missing text, repeated records, or exact duplicates explain the signal.

The controlled example shows that an alert can be technically correct while still requiring contextual interpretation. The exact duplicate pair produces a maximum Jaccard similarity of 1.0, while the summary also records the missing text and duplicate record. This supports an audit-first reading pattern: inspect data-quality counters, then pairwise distributions, then the response pair responsible for an alert.

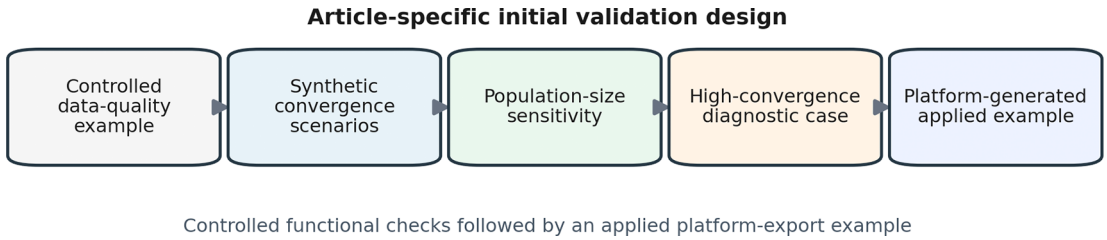

Figure 2. Article-specific initial validation design.

Table 7. Validation datasets created for this article.

Dataset	Questions	Agents	Records	Purpose
Controlled data-quality example	3	4	13	Exercises empty text, duplicate records, exact duplicates, and diverse answers.
Low-convergence scenario	3	6	18	Shows expected behaviour when responses are lexically diverse.
Moderate-convergence scenario	3	6	18	Shows expected behaviour when agents share core vocabulary but vary wording.
High-convergence scenario	3	6	18	Shows expected alert behaviour when many answers reuse the same wording.
Population-size sensitivity N = 3	2	3	6	Provides the smallest population-size sensitivity condition.
Population-size sensitivity N = 5	2	5	10	Adds pairwise opportunities while preserving comparable prompts.
Population-size sensitivity N = 10	2	10	20	Shows how pair count and vocabulary overlap change with larger N.
High-convergence diagnostic case	1	5	5	Demonstrates pair-level traceability through top_similar_pairs.

Table 8. Controlled data-quality example metrics.

Question	Raw	Valid	Missing text	Duplicates	Pairs	Max Jaccard	Distinct-2	Alert
cq1	5	4	1	1	6	1.000	0.724	True
cq2	4	4	0	0	6	0.000	1.000	False
cq3	4	4	0	0	6	0.000	1.000	False

Synthetic convergence scenarios

The convergence-scenario datasets use the same three questions and six agents. The low-convergence scenario uses diverse vocabulary and examples. The moderate-convergence scenario reuses core task vocabulary while varying sentence structure. The high-convergence scenario deliberately repeats near-identical wording by question. This controlled design checks whether the metrics respond in the expected direction when lexical overlap is manipulated directly: average and maximum Jaccard similarity increase, while lexical uniqueness decreases, as the scenario moves from low to high convergence (Figure 3; Table 9).

The comparison shows the intended separation between similarity and diversity diagnostics. In the low-convergence scenario, average and maximum Jaccard similarity remain low while lexical diversity is high. In the moderate scenario, shared vocabulary increases average overlap but remains below the provisional alert threshold. In the high scenario, repeated wording produces maximum similarity of 1.0 and a positive alert. Distinct-2 and lexical unique ratio decrease when repeated phrases dominate the response set.

Population-size sensitivity analysis

The population-size examples use two comparable questions and increase the number of agents from 3 to 5 and 10. This analysis is not intended to estimate a general scaling law. Its purpose is to demonstrate a reporting requirement: as N increases, pairs_count grows quickly, creating more opportunities for local overlap. A higher maximum similarity can therefore reflect a meaningful near-duplicate pair, a larger opportunity set, or both, which is why answers_count and pairs_count should be interpreted alongside similarity metrics (Figure 4; Table 10).

The N = 10 condition reaches the provisional maximum-similarity threshold while the smaller conditions do not. This does not mean that larger populations are less valid. It shows why maximum similarity should be interpreted alongside answers_count, pairs_count, average similarity, and diversity metrics. The lexical unique ratio also declines as more agents reuse common task vocabulary, which is expected in many prompt-controlled response sets.

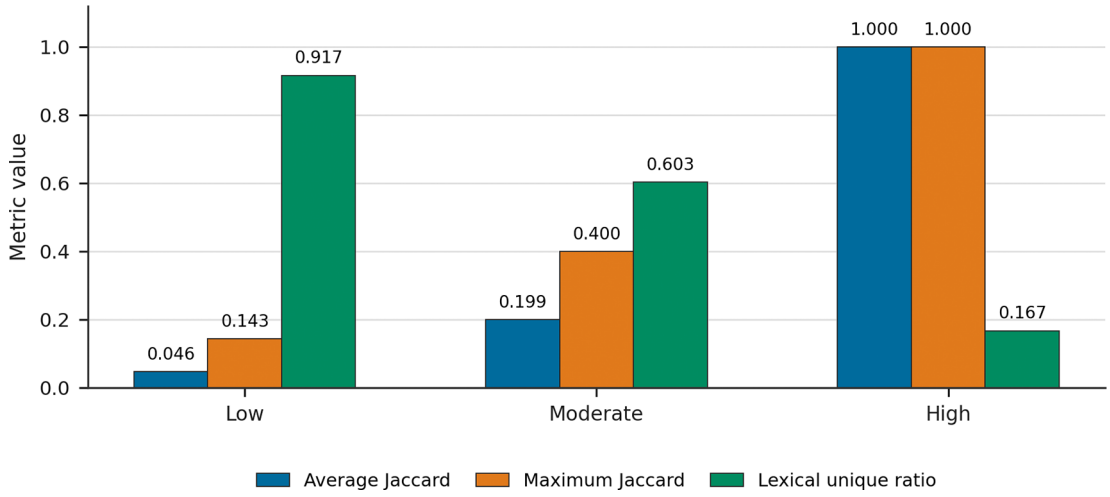

Figure 3. **Metric response across low-, moderate-, and high-convergence scenarios.**

Table 9. **Synthetic convergence scenario comparison.**

Scenario	Questions	Records	Avg. Jaccard	Max. Jaccard	Lexical unique	Distinct-2	Alert
Low	3	18	0.046	0.143	0.917	0.964	False
Moderate	3	18	0.199	0.400	0.603	0.907	False
High	3	18	1.000	1.000	0.167	0.167	True

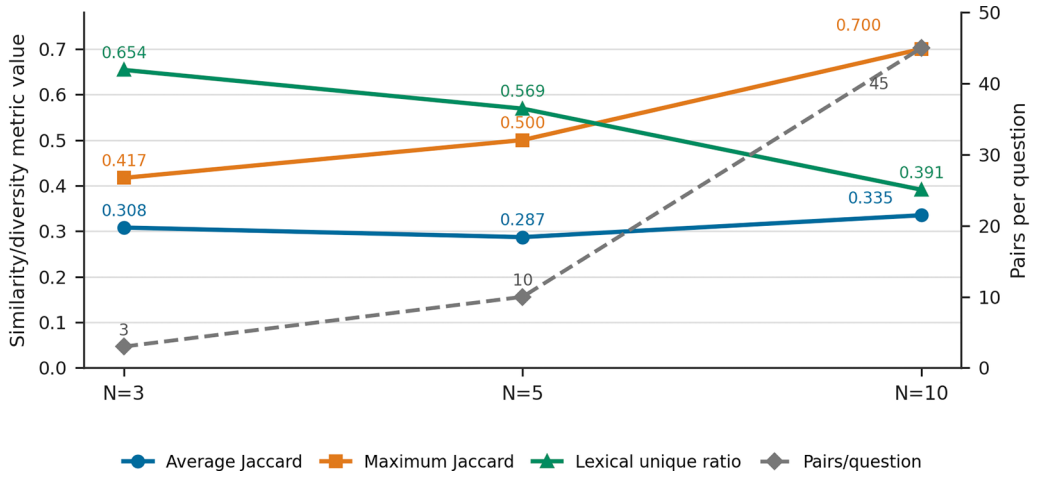

Figure 4. **Population-size sensitivity across N = 3, N = 5, and N = 10.**

Table 10. **Population-size sensitivity analysis.**

Condition	Questions	Records	Pairs/ question	Avg. Jaccard	Max. Jaccard	Lexical unique	Alert
N = 3	2	6	3	0.308	0.417	0.654	False
N = 5	2	10	10	0.287	0.500	0.569	False
N = 10	2	20	45	0.335	0.700	0.391	True

High-convergence diagnostic case

The diagnostic example contains one question and five agents. Two agents produce highly similar answers about onboarding, privacy controls, pricing notes, and support contact information; the other agents provide distinct concerns. This creates a local convergence pattern rather than a complete response-set collapse, and the pair-level output in Table 11 shows how the method makes that local pattern inspectable.

The top_similar_pairs field makes the alert auditable. Instead of reporting only that the question crossed a threshold, the output identifies the pair responsible for the highest overlap and reports both Jaccard and containment similarity. This supports manual inspection before interpreting the alert as substantive convergence.

Applied platform-generated example

The controlled synthetic examples above test expected behaviours of the method under known conditions. To complement them, we generated a small applied example from MadSynthesis and converted the export to the same platform-independent response-record schema. This example is included as an operational portability check: it shows that the method can process platform-generated synthetic data once responses are represented with question_id, agent_id, and text.

The scenario was deliberately fictional and publication-safe: a generic city-service booking app for local public events, appointments, and reminders. The run used 10 synthetic agents, three open-ended questions, English-language responses, and 30 normalized response records, with aggregate metrics reported in Table 12. The generated files contain no human respondent data, no client data, no personal data, no commercially sensitive prompt content, and no confidential scenario information. MadSynthesis is used here only as the source of a realistic platform export, not as a dependency of the method.

The reproducibility artifacts include the normalized CSV and JSONL inputs, the metric JSON output, an aggregate summary table, per-question metrics, top similar pairs, metadata, a dataset card, and a dedicated platform_generated/reproducibility_commands.md file. The per-question and top-pair tables are provided as extended data rather than expanded in the main text.

Table 11. High-convergence diagnostic pair.

Question	Top pair	Pair Jaccard	Containment	Avg. Jaccard	Max. Jaccard	Alert
dq1	a1-a2	0.833	0.909	0.093	0.833	True

Table 12. Applied platform-generated example summary.

Metric	Value
Run ID	ore-platform-20260528T095605Z-cd16d40a
Population ID	ore_platform_fictional_city_services_n10
Questions	3
Questions evaluated	3
Records total	30
Valid records total	30
Overall average Jaccard similarity	0.15
Overall maximum Jaccard similarity	0.31
Overall average lexical unique ratio	0.55
Overall average distinct-2	0.84
Overall average token entropy	7.52
Overall maximum containment similarity	0.489
Duplicate records	0
Missing text records	0
Convergence alert	False
Runtime, ms	4

In this specific synthetic run, the method did not flag potential lexical convergence under the provisional 0.70 maximum-similarity threshold. This supports operational portability to a platform export, but it should not be read as general threshold calibration, human-response validation, or an evaluation of the overall quality of MadSynthesis.

Threshold sensitivity analysis

Because the provisional 0.70 threshold governs every alert reported above, its influence can be examined directly from the deposited outputs without generating new data. Table 13 re-tabulates the per-question maximum Jaccard similarity for each validation condition and reports how many questions would be flagged at thresholds of 0.30, 0.50, 0.70, and 0.90. Within this suite, 0.70 is the only value in the range examined that flags every condition designed to be convergent, namely the high-convergence scenario, the N = 10 condition, and the diagnostic pair, while flagging no condition designed to be diverse. At 0.30 the moderate-convergence scenario, both smaller population-size conditions, and one question of the platform-generated example are flagged, because shared task vocabulary alone exceeds that value. At 0.50 the N = 5 condition is flagged at exactly 0.500. At 0.90 the designed local-convergence pair at 0.833 is missed, which is precisely the failure mode the method exists to detect. Two caveats apply. The N = 10 condition reaches exactly 0.700 and is therefore a boundary case, which reinforces the requirement to report answers_count and pairs_count alongside maximum similarity. The value of 1.000 in the controlled data-quality example is an exact-duplicate artefact rather than substantive convergence and is excluded from this reasoning, which is why data-quality counters are read before similarity metrics. This analysis is a self-consistency check on datasets constructed by the authors and not an external calibration: it does not establish that 0.70 transfers to other domains, prompt styles, languages, or model families, and calibration against human-response baselines remains future work.

CLI reproducibility check

Every validation example includes the input file and the expected JSON output generated by the command-line interface. Rather than deferring these details to the repository documentation, Table 14 maps each results table in this article to the exact input file and expected output file in the archived release, so that any single table can be regenerated independently. Each row is regenerated with the same command form, srm compute INPUT –include-pairs –language en –output OUTPUT. The generated outputs contain schema_version 0.2, the effective configuration, data-quality counters, distributional lexical metrics, and optional top similar pair diagnostics. Re-executing all nine commands against the archived inputs reproduces the deposited JSON outputs exactly, with the single exception of runtime_ms, which is retained as a diagnostic field and varies across machines and repeated executions. Because both the inputs and the expected outputs are deposited, the values reported here can be verified by direct comparison as well as by recomputation.

```
srm compute examples/ore_validation/population_size/n10.csv --include-pairs
--language en --output examples/ore_validation/population_size/n10_results.json
```

Table 13. Threshold sensitivity across the article-specific validation conditions.

Condition	Maximum Jaccard per question	Flagged at 0.30	Flagged at 0.50	Flagged at 0.70 (default)	Flagged at 0.90
Low-convergence scenario	0.062, 0.056, 0.143	0/3	0/3	0/3	0/3
Moderate-convergence scenario	0.400, 0.308, 0.385	3/3	0/3	0/3	0/3
High-convergence scenario	1.000, 1.000, 1.000	3/3	3/3	3/3	3/3
Population size N = 3	0.417, 0.385	2/2	0/2	0/2	0/2
Population size N = 5	0.500, 0.417	2/2	1/2	0/2	0/2
Population size N = 10	0.700, 0.500	2/2	2/2	1/2	0/2
High-convergence diagnostic case	0.833	1/1	1/1	1/1	0/1
Applied platform-generated example	0.310, 0.286, 0.185	1/3	0/3	0/3	0/3
Controlled data-quality example	1.000, 0.000, 0.000	1/3	1/3	1/3	1/3

Table 14. Reproducibility map from manuscript tables to archived input and expected output files.

Manuscript table	Input file under examples/ore_validation/	Expected output file
Table 8. Controlled data-quality example	controlled_example/responses.csv	controlled_example/results.json
Table 9. Low-convergence scenario	convergence_scenarios/low_convergence.csv	convergence_scenarios/low_convergence_results.json
Table 9. Moderate-convergence scenario	convergence_scenarios/moderate_convergence.csv	convergence_scenarios/moderate_convergence_results.json
Table 9. High-convergence scenario	convergence_scenarios/high_convergence.csv	convergence_scenarios/high_convergence_results.json
Table 10. Population size N = 3	population_size/n3.csv	population_size/n3_results.json
Table 10. Population size N = 5	population_size/n5.csv	population_size/n5_results.json
Table 10. Population size N = 10	population_size/n10.csv	population_size/n10_results.json
Table 11. High-convergence diagnostic pair	diagnostic_high_convergence/responses.csv	diagnostic_high_convergence/results.json
Table 12. Applied platform-generated example	platform_generated/madsynthesis_export.csv	platform_generated/madsynthesis_metrics.json

Operational overhead

The lexical computation runs locally and compares normalized response sets within each question. For a question with n_q valid responses, the method computes n_q(n_q - 1)/2 unordered response pairs, so pairwise similarity computation scales as O(n_q^2) per question, excluding the cost of normalization. Across multiple questions, the total number of pairwise comparisons is the sum of n_q(n_q - 1)/2 over all evaluated questions. In the article-specific validation examples, all runs complete at millisecond scale, but larger synthetic populations should report runtime together with answers_count and pairs_count.

Practical guidance for adoption

Researchers applying the method to a new synthetic-population study should report the response schema, normalization settings, duplicate policy, minimum answer threshold, alert threshold, software version, answers_count, and pairs_count. The raw number of records and the number of valid answers should be reported separately, because empty responses, missing identifiers, and duplicate records can otherwise make the metric denominator ambiguous. If include_pairs is enabled, the top similar pairs should be inspected before interpreting an alert as substantive convergence; Table 15 links these reporting checks to common interpretation patterns.

Table 15. Practical interpretation guide for common diagnostic patterns.

Situation	Diagnostic signal	Recommended interpretation
High max similarity, low average similarity	One or a few top_similar_pairs dominate the alert.	Inspect local convergence; do not assume the whole population collapsed.
High containment similarity, moderate Jaccard	A short answer is largely contained in a longer answer.	Check whether one response paraphrases or extends another.
Low distinct-2 or low token entropy	Vocabulary or phrase patterns are concentrated.	Review prompts for repeated wording or overly narrow answer framing.
insufficient_data status	Too few valid answers after normalization.	Report the exclusion reason and avoid interpreting pairwise metrics for that question.
Duplicate records present	duplicate_records_count is greater than zero.	Declare the duplicate policy and verify that repeated records were handled as intended.
Higher max similarity with larger N	pairs_count increases as more agents are included.	Report denominators and avoid comparing max values without population-size context.

Discussion

The method contributes an observability layer for synthetic-population research. It does not attempt to replace qualitative interpretation, human validation, semantic similarity assessment, or downstream task evaluation. Instead, it provides a reproducible signal for a narrower question: are generated agents producing responses that are lexically too similar for a given prompt, population, model, and configuration?

This framing matters because convergence can be local. A response set may have low average similarity while still containing one pair of near-duplicate answers. Maximum Jaccard similarity is therefore included alongside average similarity, and distributional summaries such as median, p90, and standard deviation make the pairwise pattern more visible. Conversely, vocabulary diversity can remain high even when semantic content is aligned, which is why lexical unique ratio and distinct-n metrics should not be interpreted as complete validity scores.

The method also relates to wider concerns about homogenization and model collapse. Those concerns are often discussed at the level of training data, creative output, or model ecosystems. The present method addresses a more operational layer: during a synthetic-population study, the researcher needs to know whether the specific generated answer set contains repeated wording or local near-duplicates. This local view does not solve the broader problem, but it makes one observable failure mode easier to report.

Population size is a key interpretive factor. Because the method compares pairs within each question, the number of possible pairs increases quickly as N grows. For N = 3 there are three pairs per question, for N = 5 there are ten, and for N = 10 there are forty-five. A high maximum similarity is more likely to appear when there are more pairs. For this reason, max_jaccard_similarity should not be compared across runs without reporting answers_count and pairs_count.

The main limitation of the current version is its lexical scope, and we state its consequence plainly: the method cannot detect semantic convergence. Two answers can express the same opinion with different words, producing low lexical overlap despite semantic convergence, and two answers can also reuse words while making different claims. Partial coverage exists but should not be overstated: max_containment_similarity exposes subset-like overlap when response lengths differ, and distinct_2 exposes repeated short phrases and answer templates. Neither is a measure of meaning-level agreement, so a synthetic population can pass every check reported here while remaining semantically homogeneous. This limitation is deliberate because lexical metrics are transparent, inexpensive, dependency-free, and reproducible without model access. The method establishes a lexical baseline before introducing more complex metrics that may depend on embedding models, parsers, or human-labelled reference data.

The second limitation is validation scale. The examples in this manuscript are controlled synthetic examples created to test expected behaviour, reproducibility, and diagnostic interpretability. They do not establish general-purpose thresholds, and they do not compare synthetic response distributions against human-response baselines. Future validation should include larger populations, additional task types, different model families, domain-specific prompts, and expert assessment of whether flagged response pairs represent excessive convergence. The applied platform-generated example adds an operational portability check, but it remains synthetic and scenario-specific; validation with real domains, human-response baselines, or external benchmarks remains future work.

A third limitation concerns what the method observes. It measures dispersion of the generated outputs, not fidelity of the conditioning. It does not verify that responses actually reflect the demographic, behavioural, or preference attributes assigned to each agent. A response set can be lexically diverse, and pass every check reported here, while ignoring the configured personas entirely, because diversity of wording and faithfulness to a configured profile are different properties. Conversely, a response set can be lexically convergent precisely because the configured differences are irrelevant to the question that was asked. Verifying conditioning fidelity requires per-agent attribute metadata and attribute-conditional comparison of response dispersion, which the response-record schema anticipates through the optional population_id field but does not yet exploit.

A fourth limitation is that the method depends on responses being expressible as question_id, agent_id, and text. Three mitigations reduce this constraint in practice: the loader accepts common field aliases, three input formats are supported, and a dedicated loader handles interaction-style exports including runs in which a single interviewer question applies to all agents. The residual limitation is nevertheless real. A fully black-box platform that offers no export capability requires a small adapter, in practice a single function mapping its output onto the three required fields. The method is platform-independent, but it is not export-independent.

The default threshold of 0.70 is therefore provisional. The threshold sensitivity analysis reported in Table 13 shows that, within this validation suite, 0.70 is the only value in the range examined that flags every condition designed to be

Table 16. Planned extensions beyond the current lexical method.

Extension	Motivation	Expected requirement
Semantic similarity	Flag meaning-level convergence when wording differs.	Optional embedding model and documented model/version.
Syntactic-template monitoring	Identify repeated answer structures or rhetorical templates.	Tokenizer/parser or lightweight pattern extractor.
Human-baseline calibration	Interpret synthetic dispersion relative to observed human response variation.	Human or benchmark response dataset with ethics approval where needed.
Longitudinal monitoring	Track convergence across repeated runs or model updates.	Stable run metadata and archived output payloads.
Agent-conditioning fidelity	Verify whether responses vary with the configured agent attributes, not only with wording.	Per-agent attribute metadata and attribute-conditional dispersion comparison.

convergent while flagging no condition designed to be diverse: at 0.30 the diverse conditions are over-flagged, at 0.50 the N = 5 condition is flagged at exactly 0.500, and at 0.90 the designed local-convergence pair at 0.833 is missed. That is a self-consistency check on datasets constructed by the authors rather than an external calibration, and it does not establish that 0.70 transfers to other domains, prompt styles, languages, or model families. The threshold is useful because it is explicit and easy to audit, but it should be treated as a starting point for inspection rather than a stable validity boundary. In some domains, lower thresholds may be useful because repeated phrasing is problematic; in others, higher thresholds may be appropriate because task wording naturally constrains vocabulary.

The planned extension path includes semantic similarity, syntactic-template monitoring, and human-baseline calibration (Table 16). Semantic similarity can flag potential convergence when agents use different vocabulary to express similar meanings, but it requires model and version documentation for embedding-based methods (Reimers and Gurevych, 2019; Zhang et al., 2020). Syntactic-template monitoring can identify repeated answer structures (Shaib et al., 2024). Human-baseline calibration can help determine whether synthetic dispersion is narrower, broader, or differently shaped than observed human response variation.

Conclusions

This Method Article presents a lightweight computational method for monitoring lexical response convergence in LLM-based synthetic populations. The method uses a generic response-record schema, deterministic normalization, interpretable lexical and diversity metrics, data-quality counters, threshold-based inspection flags, optional pair-level diagnostics, and reproducible JSON outputs.

The central methodological claim is that response dispersion should be treated as an explicit observability problem in synthetic-population research. Synthetic agents should not be evaluated only by fluency or aggregate thematic plausibility. Researchers also need simple, reproducible ways to inspect whether generated responses preserve variation across configured agents. The proposed method provides a practical lexical baseline toward that goal.

The article-specific examples provide initial validation under controlled synthetic conditions, and we state their evidential status precisely. They demonstrate that the metrics respond in the expected direction under manipulated lexical overlap, that the data-quality counters surface missing text, repeated records, and exact duplicates before metric interpretation, that alerts are traceable to the response pair responsible for them, that population size changes the number of pairwise opportunities for local overlap, and that the same input schema processes a platform-generated export. They do not demonstrate a generalizable threshold, agreement with human-response baselines, semantic validity, or fidelity of agent conditioning. The method should therefore be adopted as a reproducible lexical monitoring layer that runs before more expensive validation, and not as evidence that a synthetic population is adequately diverse. The next step is to calibrate the method against human-response baselines and domain-specific expert judgments.

Ethics and consent

This manuscript does not report data from human participants or animals. The initial validation uses synthetic response records created for software and methodological evaluation. Ethical approval and consent were therefore not required for the current synthetic-output analysis. If future work uses human-response baselines, the relevant ethics approval, consent process, and data-protection safeguards will be reported.

Data availability

Source data

No human-participant source data are reported in this manuscript. The initial validation uses article-specific synthetic response datasets created to exercise controlled convergence, data-quality handling, population-size sensitivity, and pair-level diagnostic behaviour. No personally identifiable participant data are included. The applied platform-generated example is also synthetic, was generated specifically for this manuscript, and uses a fictional scenario.

Underlying data

All underlying data are openly available inside the archived release of synthetic-response-metrics v0.2.5 at https://doi.org/10.5281/zenodo.20537552, in the directory examples/ore_validation/. The archived record is openly accessible under the MIT License, is not embargoed, and requires no access request. Both the validation inputs and the expected metric outputs are deposited, so every value reported in this article can be verified by direct comparison as well as by recomputation. The data are deposited together with the software rather than as a separate record because they are small, fully synthetic, and version-locked to the implementation that produces the expected outputs; separating them would break the input, version, and output coupling that makes the reported values verifiable. The Zenodo concept DOI for the software record is https://doi.org/10.5281/zenodo.20440939.

The deposited directory examples/ore_validation/contains 28 files. The validation inputs and their expected outputs are controlled_example/responses.csv and controlled_example/results.json; convergence_scenarios/low_convergence.csv and convergence_scenarios/low_convergence_results.json; convergence_scenarios/moderate_convergence.csv and convergence_scenarios/moderate_convergence_results.json; convergence_scenarios/high_convergence.csv and convergence_scenarios/high_convergence_results.json; population_size/n3.csv and population_size/n3_results.json; population_size/n5.csv and population_size/n5_results.json; population_size/n10.csv and population_size/n10_results.json; and diagnostic_high_convergence/responses.csv and diagnostic_high_convergence/results.json. The applied platform-generated example additionally includes platform_generated/madsynthesis_raw_export.json, platform_generated/madsynthesis_export.csv, platform_generated/madsynthesis_export.jsonl, platform_generated/madsynthesis_metrics.json, platform_generated/madsynthesis_summary.csv, platform_generated/madsynthesis_by_question.csv, platform_generated/madsynthesis_top_pairs.csv, and platform_generated/metadata.json. The documentation files deposited alongside the data are ore_validation/README.md, platform_generated/README.md, platform_generated/dataset_card.md, and platform_generated/reproducibility_commands.md. Table 7 reports the question, agent, and record counts for each dataset, and Table 14 maps each results table in this article to its input file and expected output file.

Extended data

Extended data are archived with the same versioned release at https://doi.org/10.5281/zenodo.20537552. They include README files, dataset cards, reproducibility commands, normalized CSV/JSONL inputs, expected JSON outputs, summary CSV files, per-question outputs, and top-pair diagnostic files. The complete English and Spanish stopword lists used by the default normalization pipeline, comprising 20 and 21 terms respectively and 41 terms in combined automatic mode, are provided in the archived source file src/synthetic_response_metrics/normalization.py, because convergence scores depend on them. Manuscript tables are derived from these archived inputs and outputs.

Software availability

Reference implementation: synthetic-response-metrics. Version: 0.2.5. Public repository: https://github.com/ralvarezmartin/synthetic-response-metrics. Version tag: v0.2.5. Archived software release DOI: https://doi.org/10.5281/zenodo.20537552. Repository license: MIT. The archived release includes source code, tests, README, LICENSE, pyproject.toml, CITATION.cff, MANIFEST.in, validation examples, normalized inputs, and expected metric outputs.

Installation from the archived or tagged repository release can be performed by cloning https://github.com/ralvarezmartin/synthetic-response-metrics, checking out v0.2.5, and running pip install -e. from the repository root. Main command-line entry point: srm compute. Example command: srm compute examples/ore_validation/population_size/n10.csv --include-pairs --language en --output examples/ore_validation/population_size/n10_results.json.

Reproducibility

The reference implementation requires Python 3.10 or later and declares no third-party runtime dependencies, so the reported metrics can be recomputed with a standard Python installation alone. The archived release can be installed by checking out tag v0.2.5 and running pip install -e. from the repository root, and it can also be executed without installation by setting PYTHONPATH to src and invoking python -m synthetic_response_metrics.cli. Table 14 lists, for every results table in this article, the exact input file and expected output file in the archived release. Re-executing all nine commands against the archived inputs reproduces the deposited JSON outputs exactly, with the single exception of runtime_ms,

which varies across machines and repeated executions. The archived release also includes 17 automated tests, run with pytest from the repository root. Runtime values may vary across machines, but the reported lexical metrics are generated deterministically from the archived inputs and package version 0.2.5.

Reporting guidelines

No clinical, biomedical, animal-research, or human-participant reporting guideline is directly applicable to this computational Method Article. Reproducibility details are provided through the input schema, normalization parameters, configuration validation rules, metric definitions, command-line interface, test cases, software license, archived repository release, example data files, expected outputs, generated tables, and reproducibility commands. If future validation includes human-response baselines, the appropriate reporting and ethics requirements will be applied to that study.

Acknowledgements

This research has been supported by the project "A catalyst for EuropeaN ClOUd Services in the era of data spaces, high-performance and edge computing (NOUS)", Grant Agreement Number 101135927. Funded by the European Union, views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

This article was made possible by the results of the MADSYNTHESIS AI project (TSI-100130-2024-0134), funded by the European Union – Next Generation EU, as part of the Recovery, Transformation, and Resilience Plan, through the Sectoral Data Spaces Program of the Ministry for Digital Transformation and the Civil Service.

During the preparation of this manuscript, ChatGPT was used for formatting support to adapt the manuscript to the journal template. All authors reviewed, edited, and approved the final manuscript and take full responsibility for its content.

References

Aher GV, Arriaga RI, Kalai AT: **Using Large Language Models to Simulate Multiple Humans and Replicate Human Subject Studies.** *Proceedings of Machine Learning Research.* 2023; **202**: 337–371.
Reference Source

Alvarez R, Soroberto M, De la Prieta F, *et al.*: **synthetic-response-metrics: A lightweight Python package for monitoring lexical response convergence in synthetic populations.** *Zenodo.* 2026.
Publisher Full Text

Anderson BR, Shah JH, Kreminski M: **Homogenization effects of large language models on human creative ideation.** *Proceedings of the 16th Conference on Creativity & Cognition.* 2024; 413–425.
Publisher Full Text

Argyle LP, Busby EC, Fulda N, *et al.*: **Out of One, Many: Using Language Models to Simulate Human Samples.** *Polit. Anal.* 2023; **31**(3): 337–351.
Publisher Full Text

Arora N, Chakraborty I, Nishimura Y: **AI-Human Hybrids for Marketing Research: Leveraging Large Language Models as Collaborators.** *J. Mark.* 2025; **89**(2): 43–70.
Publisher Full Text

Chang Y, Wang X, Wang J, *et al.*: **A Survey on Evaluation of Large Language Models.** *ACM Trans. Intell. Syst. Technol.* 2024; **15**(3): 1–45.
Publisher Full Text

De Freitas J, Nave G, Puntoni S: **Ideation with Generative AI-in Consumer Research and Beyond.** *J. Consum. Res.* 2025; **52**(1): 18–31.
Publisher Full Text

Doshi AR, Hauser OP: **Generative AI enhances individual creativity but reduces the collective diversity of novel content.** *Sci. Adv.* 2024; **10**.
PubMed Abstract | Publisher Full Text | Free Full Text

Guo T, Chen X, Wang Y, *et al.*: **Large Language Model based Multi-Agents: A Survey of Progress and Challenges.** *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence.* 2024a; 8048–8057.
Publisher Full Text

Guo Y, Shang G, Vazirgiannis M, *et al.*: **The Curious Decline of Linguistic Diversity: Training Language Models on Synthetic Text.** *Findings of the Association for Computational Linguistics: NAACL.* 2024b; **2024**: 3589–3604.
Publisher Full Text

Holtzman A, Buys J, Du L, *et al.*: **The Curious Case of Neural Text Degeneration.** *International Conference on Learning Representations.* 2020.
Reference Source

Jaccard P: **Étude comparative de la distribution florale dans une portion des Alpes et du Jura.** *Bulletin de la Société Vaudoise des Sciences Naturelles.* 1901; **37**: 547–579.

Li J, Galley M, Brockett C, *et al.*: **A Diversity-Promoting Objective Function for Neural Conversation Models.** *Proceedings of NAACL-HLT.* 2016; **2016**: 110–119.
Publisher Full Text

Manning CD, Raghavan P, Schutze H: *Introduction to Information Retrieval.* Cambridge: Cambridge University Press; 2008.
Publisher Full Text

Mohammadi M, Li Y, Lo J, *et al.*: **Evaluation and Benchmarking of LLM Agents: A Survey.** *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25).* 2025.
Publisher Full Text

Papineni K, Roukos S, Ward T, *et al.*: **BLEU: a Method for Automatic Evaluation of Machine Translation.** *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics.* 2002; 311–318.
Publisher Full Text

Park JS, O'Brien J, Cai CJ, *et al.*: **Generative Agents: Interactive Simulacra of Human Behavior.** *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology.* 2023; 1–22. Article 2.
Publisher Full Text

Reimers N, Gurevych I: **Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks.** *Proceedings of EMNLP-IJCNLP.* 2019; **2019**: 3982–3992.
Publisher Full Text

Sandve GK, Nekrutenko A, Taylor J, *et al.*: **Ten Simple Rules for Reproducible Computational Research.** *PLoS Comput. Biol.* 2013; **9**(10): e1003285.
PubMed Abstract | Publisher Full Text | Free Full Text

Shaib C, Elazar Y, Li JJ, *et al.*; **Detection and Measurement of Syntactic Templates in Generated Text.** *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing.* 2024; 6416–6431.
Publisher Full Text

Shannon CE: **A Mathematical Theory of Communication.** *Bell Syst. Tech. J.* 1948; **27**(3): 379–423. 27(4):623–656.
Publisher Full Text

Shumailov I, Shumaylov Z, Zhao Y, *et al.*: **AI models collapse when trained on recursively generated data.** *Nature.* 2024; **631**: 755–759.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Smith AM, Katz DS, Niemeyer KE, *et al.*: **Software Citation Principles.** *PeerJ Computer Science.* 2016; **2**: e86.
Publisher Full Text

Toubia O, Gui GZ, Peng T, *et al.*: **Database Report: Twin-2K-500: A Data Set for Building Digital Twins of over 2,000 People Based on Their Answers to over 500 Questions.** *Mark. Sci.* 2025; **44**(6): 1446–1455.
Publisher Full Text

Wilkinson MD, Dumontier M, Aalbersberg IJ, *et al.*: **The FAIR Guiding Principles for scientific data management and stewardship.** *Sci. Data.* 2016; **3**: 160018.
Publisher Full Text

Wilson G, Bryan J, Cranston K, *et al.*: **Good Enough Practices in Scientific Computing.** *PLoS Comput. Biol.* 2017; **13**(6): e1005510.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Zhang S, Xu J, Alvero AJ: **Generative AI Meets Open-Ended Survey Responses: Research Participant Use of AI and Homogenization.** *Sociol. Methods Res.* 2025; **54**(3): 1197–1242.
Publisher Full Text

Zhang T, Kishore V, Wu F, *et al.*: **BERTScore: Evaluating Text Generation with BERT.** *International Conference on Learning Representations.* 2020.
Reference Source

Zhu Y, Lu S, Zheng L, *et al.*: **Texygen: A Benchmarking Platform for Text Generation Models.** *Proceedings of SIGIR.* 2018; **2018**: 1097–1100.
Publisher Full Text

Open Peer Review

Current Peer Review Status: ? ?

Version 2

Reviewer Report 11 September 2026

https://doi.org/10.21956/openreseurope.26831.r77929

© **2026 Illiashenko O.** This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

?

Oleg Illiashenko
Leeds Beckett University, Leeds, UK

The article presents a lightweight, deterministic method for monitoring lexical response convergence in LLM-based synthetic populations.
The content and the methodology are clear: synthetic agents configured as different participants may still converge towards similar wording, examples, and response templates, so researchers need an inexpensive observability layer before investing in semantic, human-baseline, or domain-specific validation. The method description is technically fair for the stated purpose, because the authors correctly scope the contribution as lexical rather than semantic convergence detection.
The paper is reproducible and has link to the public repo, which is also good.

The controlled examples are good for a method article and show expected behaviour: low-, moderate-, and high-convergence scenarios move in the expected direction, the N=10 case demonstrates pair-count sensitivity, and the diagnostic example traces an alert to a specific highly similar pair.
However, the conclusions about “performance” aren’t supported in full: the validation remains author-constructed, synthetic, small-scale, and self-consistency-oriented rather than externally calibrated against real human responses, independent synthetic-population platforms, multiple LLM families, or expert-labelled convergence cases. The authors do acknowledge this limitation, and it shows that the authors respect it without overclaiming.

The main correction from my side will be the recommendations:
(a) to ensure that every performance-related statement remains explicitly limited to lexical monitoring under controlled synthetic examples,
(b) any implication that the method validates synthetic-population diversity, representativeness, persona fidelity, or semantic independence must be removed or toned down; (c) the provisional 0.70 threshold should be described only as an inspection heuristic, not as a validity boundary, because the threshold sensitivity analysis is based on datasets designed by the authors themselves.
The paper would benefit from an explanation that the “auto” language setting is not genuine language detection but a combined stop-word mode, because this could mislead possible users in

applying the method to multilingual or non-Latin datasets.

Overall, the article is scientifically good as a narrowly scoped computational method article.

Is the rationale for developing the new method (or application) clearly explained?
Yes

Is the description of the method technically sound?
Yes

Are sufficient details provided to allow replication of the method development and its use by others?
Yes

If any results are presented, are all the source data underlying the results available to ensure full reproducibility?
Yes

Are the conclusions about the method and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: cyber security, functional safety, AI, IoT, LLM, UAV

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Version 1

Reviewer Report 21 July 2026

https://doi.org/10.21956/openreseurope.26377.r76768

© 2026 OLUWADAYISI A. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

?

AKIN OLAWALE OLUWADAYISI
COLLEGE OF LAW, Wesley University of Science and Technology Ondo (Ringgold ID: 398081), Ondo, Ondo, Nigeria

SUMMARY OF WORK
1.0 This work introduces a practical computational method for monitoring response convergence within LLM-generated synthetic populations. The author's approach utilizes a standardized response schema, deterministic text normalization, lexical and diversity metrics, data quality counters, threshold-based alerts, optional pair-level diagnostics, and reproducible JSON outputs.
2.0 The primary contribution lies in conceptualizing response dispersion as an observability challenge. It posits that synthetic agents require evaluation not only for fluency and thematic plausibility but also for preserving meaningful variation. The proposed method provides a foundational lexical assessment for this verification.
3.0 Controlled synthetic experiments demonstrate the method's efficacy in detecting artificial convergence, identifying data quality issues, illustrating population size effects, and pinpointing specific response pairs contributing to overlap. The method is applicable to both designed test cases and normalized outputs from synthetic platforms adhering to the specified schema. Future research is recommended to calibrate thresholds and metrics against human response data and expert judgment.

SPECIFIC COMMENTS ON (Q2-Q5)
4.0 A significant limitation of the work is the method's focus on lexical features, potentially overlooking semantic convergence where distinct vocabulary masks identical meanings, opinions, or reasoning patterns, a known LLM behavior. Further reasons for the "No" and "Partly" response include:
4.1 ***Preliminary validation:*** The current demonstrations are confined to controlled synthetic conditions. It is believed that real-world synthetic populations, involving complex prompts and contexts, may exhibit different behaviors, necessitating grounding in human baselines.
4.2 ***Threshold sensitivity:*** The reliance on threshold-based flags introduces potential arbitrariness and requires calibration to avoid excessive noise or missed homogenization.
4.3 ***Schema dependency:*** Applicability is contingent on the availability of a standardized input schema, potentially limiting use with ad-hoc or black-box LLM platforms.
4.4 Shallow observability: While enhancing inspection, the method used by the author does not fully address whether model-generated responses respect configured agent differences beyond lexical variation.

In essence, this method serves as a valuable initial diagnostic tool, offering transparency but not a complete solution for ensuring genuine population diversity. Comprehensive evaluation would likely necessitate extensions incorporating semantic or embedding-based analyses and stronger empirical validation against human behavior.

References
1. Alvarez R, Soroberto M, De la Prieta F, Cartón M: A lightweight computational method for monitoring response convergence in LLM-based synthetic populations. *Open Research Europe*. 2026; **6**. Publisher Full Text

Is the rationale for developing the new method (or application) clearly explained?
Yes

Is the description of the method technically sound?
Partly

Are sufficient details provided to allow replication of the method development and its use by others?
No

If any results are presented, are all the source data underlying the results available to ensure full reproducibility?
No

Are the conclusions about the method and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Jurisprudence and International Law, Advanced Research methodology, Cybersecurity, AI and Human Rights.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 30 Jul 2026

Rodrigo Álvarez Martín

Dear Akin,
Thank you for a careful and constructive report. Your comments identified two areas where the manuscript under-specified work that had in fact been completed and deposited, and three areas where our statement of scope was too weak. We have revised the article accordingly. Below we respond to each point and indicate where the change was made.

ON THE LEXICAL SCOPE OF THE METHOD (COMMENT 4.0)
We accept this criticism and have made the scope explicit rather than implicit. The title now reads "monitoring lexical response convergence", aligning it with the abstract. The Method assumptions subsection and the Discussion now state directly that the method cannot detect semantic convergence: two agents expressing the same opinion in different vocabulary will produce a low Jaccard score and will not be flagged. We also clarify what is only partially covered, namely max_containment_similarity for subset-like overlap and distinct_2 for repeated short phrases and answer templates, while stating that neither measures meaning-level agreement and that a synthetic population can pass every check we report while remaining semantically homogeneous. We have not added embedding-based metrics in this version, and we want to be transparent about why. The implementation deliberately declares no third-party runtime dependencies and computes every metric with the Python standard library, which is what allows the reported values to be recomputed indefinitely without drift from external library versions. Adding an embedding layer would trade that property for coverage we cannot yet validate. Semantic similarity therefore remains listed as a planned extension, with its model and version

documentation requirement stated.

ON PRELIMINARY VALIDATION AND HUMAN BASELINES (COMMENT 4.1)
We agree. The Conclusions now separate what the examples demonstrate from what they do not. Demonstrated: expected metric direction under manipulated lexical overlap; data-quality counters surfacing missing text, repeated records and exact duplicates before metric interpretation; traceability of alerts to the responsible response pair; the effect of population size on the number of pairwise opportunities; and operational portability to a platform-generated export. Not demonstrated: generalizable threshold calibration, agreement with human-response baselines, semantic validity, or fidelity of agent conditioning. We have removed framing that implied broader validation than the controlled examples support.

ON THRESHOLD SENSITIVITY (COMMENT 4.2)
This prompted a substantive addition. The revised article includes a new threshold sensitivity analysis, Table 13, reporting for every validation condition how many questions would be flagged at thresholds of 0.30, 0.50, 0.70 and 0.90. The values are re-tabulated from the per-question maxima in the already-deposited JSON outputs, so the analysis introduces no new data and can be checked directly against the archive. The analysis shows that at 0.30 the conditions designed to be diverse are over-flagged, because shared task vocabulary alone exceeds that value; that at 0.50 the N = 5 condition is flagged at exactly 0.500; that at 0.70 exactly the conditions designed to be convergent are flagged and no others; and that at 0.90 the designed local-convergence pair at 0.833 is missed, which is precisely the failure mode the method exists to detect. We also now state explicitly that the N = 10 condition reaches exactly 0.700 and is therefore a boundary case, which reinforces the requirement to report answers_count and pairs_count alongside maximum similarity. We are careful not to overclaim this is a self-consistency check on datasets we constructed, not an external calibration, and the default threshold remains provisional pending calibration against human-response baselines.

ON SCHEMA DEPENDENCY (COMMENT 4.3)
We accept the point and have expanded the Input data model subsection and the Discussion. Version 1 already documented two mitigations that are present in the implementation: acceptance of common field aliases (question, content, answer, response, agentId, participant_id) and support for CSV, JSON and JSONL. A third was implemented but not described, and is now stated: a dedicated loader for interaction-style exports, including runs in which a single interviewer question applies to all agents. What version 1 did not do was frame any of this as a response to the schema-dependency question, which the revision now does explicitly. We also state the residual limitation plainly: a fully black-box platform that offers no export capability requires a small adapter, in practice a single function mapping its output onto the three required fields. The method is platform-independent, but it is not export-independent.

ON SHALLOW OBSERVABILITY (COMMENT 4.4)
We consider this the most important conceptual gap you identified, and it was not addressed in version 1. The Discussion now distinguishes dispersion of the outputs from fidelity of the conditioning: the method does not verify that responses reflect the attributes

assigned to each agent, and a response set can be lexically diverse while ignoring the configured personas entirely, because diversity of wording and faithfulness to a configured profile are different properties. Conversely, a set can be lexically convergent precisely because the configured differences are irrelevant to the question asked. We have added Agent-conditioning fidelity as a planned extension in Table 16, noting that the response-record schema already carries the optional population_id field as the natural hook for attribute-conditional comparison.

ON REPLICATION DETAILS (FORM QUESTION 3)
You were right that the manuscript was under-specified: version 1 deferred reproduction detail to the repository documentation and included only two example commands in the text. The revision brings this into the article. The Reference implementation subsection and the Reproducibility section now state the computational environment: Python 3.10 or later, MIT licence, and no third-party runtime dependencies, since every metric is computed with the Python standard library. They also give the installation path and the no-install invocation path, the module layout behind the public API, and the 17 automated tests with the behaviours they cover. The Normalization pipeline subsection now documents the normalization contract precisely: the exact cleanup pattern, the fact that accented Latin characters are preserved, the ordering of the token-length and stopword filters, and the stopword list sizes, which are 20 English terms, 21 Spanish terms and 41 terms in combined automatic mode. The lists are deliberately minimal and ship with the package rather than being inherited from an external NLP distribution, so we now provide them explicitly, because convergence scores depend on them. Finally, a new reproducibility map, Table 14, gives for every results table in this article the exact input file and expected output file in the archived release, so that any single table can be regenerated independently. We re-executed all nine commands against the archived inputs while preparing this revision: they reproduce the deposited JSON outputs exactly, with the single exception of runtime_ms, which is machine-dependent and already declared as such.

ON AVAILABILITY OF SOURCE DATA (FORM QUESTION 4)
We would like to clarify the status of the deposit, and we have also made it far easier to verify. All underlying data were, and are, openly deposited. The archived release at https://doi.org/10.5281/zenodo.20537552 contains 28 files under examples/ore_validation/, comprising every validation input and every expected metric output. The record is open under the MIT Licence, is not embargoed, and requires no access request. We recognise, however, that this was not discoverable from the article, nor from the Zenodo landing page, which lists a single archive file. The Data availability section has therefore been rewritten to enumerate the complete file manifest with the path of every file inside the archive, to cross-reference the dataset counts already given in Table 7, and to state explicitly that expected outputs are deposited alongside inputs so that reported values can be verified by direct comparison as well as by recomputation. We also now explain why the data are deposited with the software rather than separately: they are small, fully synthetic, and version-locked to the implementation that produces the expected outputs, and separating them would break the input, version and output coupling that makes the values verifiable. If you would prefer the validation data to carry an independent data DOI, we are willing to create one.

SUMMARY OF WHAT REMAINS OPEN

We want to be clear about what this version does not resolve. It does not add semantic or embedding-based metrics, it does not calibrate the threshold against human-response baselines, and it does not verify agent-conditioning fidelity. These are stated as limitations and listed as planned extensions rather than presented as addressed. What the revision does provide is an honestly scoped lexical baseline whose every reported number can now be reproduced from the article itself. Thank you again for the review, which improved the article materially.

Regards,
The authors

Competing Interests: No competing interests were disclosed.